A Report on GRASP

- Mubasshira Musarrat
- 1905088

The Greedy Randomized Adaptive Search Procedure (GRASP) algorithm is a metaheuristic optimization technique used to solve combinatorial optimization problems. The grasp algorithm has two phases: 1) construction phase & 2) local search phase. The runtime for this algorithm is $O(n^3)$.

For the construction phase we can select random, greedy or semi-greedy methods. In our given assignment we had to determine the max-cut value of a given graph implementing all three methods & compare them with the value calculated from grasp. We were given a dataset consisting of 54 graphs and known best values for 24 of them.

Table 1:

Name	Problem Size		Simple Randomized or Randomized-1	Simple Greedy or Greedy-1	Semi-greedy-1 α = 0.5	Simple Local or Local-1		Grasp-1		Known Best Solution or Upper Bound
	|V| or n	|E| or m				Number of Iterations	Best Value	Number of Iterations	Best Value	
G1	800	19176	11021	11260	11120	73	11302	50	11371	12078
G2	800	19176	11022	11273	11130	71	11311	50	11371	12084
G3	800	19176	11020	11258	11125	69	11299	50	11365	12077
G4	800	19176	11020	11273	11125	71	11310	50	11388	
G5	800	19176	11027	11284	11130	71	11309	50	11378	
G6	800	19176	1549	1763	1582	84	1806	50	1867	
G7	800	19176	1373	1591	1429	83	1658	50	1777	
G8	800	19176	1380	1613	1427	85	1658	50	1736	
G9	800	19176	1423	1651	1482	82	1703	50	1810	
G10	800	19176	1389	1602	1419	84	1650	50	1764	
G11	800	1600	410	483	414	15	445	50	462	627
G12	800	1600	395	474	401	16	432	50	464	621
G13	800	1600	417	495	420	17	455	50	484	645
G14	800	4694	2868	2943	2929	27	2963	50	2986	3187

Name	Problem Size		Simple Randomized or Randomized-1	Simple Greedy or Greedy-1	Semi-greedy-1 α = 0.5	Simple Local or Local-1		Grasp-1		Known Best Solution or Upper Bound
	|V| or n	|E| or m				Number of Iterations	Best Value	Number of Iterations	Best Value	
G15	800	4661	2849	2921	2913	25	2945	50	2964	3169
G16	800	4672	2853	2925	2920	25	2952	50	2974	3172
G17	800	4667	2851	2923	2913	27	2947	50	2965	
G18	800	4694	696	842	721	63	830	50	894	
G19	800	4661	613	753	628	66	741	50	798	
G20	800	4672	646	786	670	62	775	50	817	
G21	800	4667	635	773	648	68	763	50	808	
G22	2000	19990	12356	12793	12642	113	12854	50	12943	14123
G23	2000	19990	12357	12795	12631	115	12850	50	12936	14129
G24	2000	19990	12347	12791	12630	118	12853	50	12940	14131
G25	2000	19990	12354	12794	12630	117	12853	50	12932	
G26	2000	19990	12351	12785	12625	111	12831	50	12882	
G27	2000	19990	2319	2699	2393	175	2746	50	2826	
G28	2000	19990	2295	2657	2350	173	2700	50	2780	
G29	2000	19990	2376	2752	2439	174	2790	50	2873	
G30	2000	19990	2370	2753	2445	176	2796	50	2883	
G31	2000	19990	2294	2659	2352	180	2717	50	2822	
G32	2000	4000	1017	1188	1028	38	1104	50	1152	1560
G33	2000	4000	976	1183	992	41	1076	50	1132	1537
G34	2000	4000	976	1176	984	42	1068	50	1110	1541
G35	2000	11778	7186	7380	7360	61	7437	50	7471	8000
G36	2000	11766	7175	7378	7353	60	7431	50	7468	7996
G37	2000	11785	7193	7388	7360	61	7437	50	7465	8009
G38	2000	11779	7183	7376	7356	62	7435	50	7464	
G39	2000	11778	1683	2014	1693	170	1984	50	2047	
G40	2000	11766	1648	2005	1707	161	1983	50	2064	
G41	2000	11785	1666	2022	1707	161	1988	50	2082	
G42	2000	11779	1746	2094	1771	167	2063	50	2165	
G43	1000	9990	6174	6378	6289	62	6406	50	6452	7027
G44	1000	9990	6162	6383	6288	60	6402	50	6454	7022
G45	1000	9990	6173	6382	6280	62	6399	50	6445	7020
G46	1000	9990	6176	6387	6294	61	6409	50	6472	
G47	1000	9990	6179	6393	6295	63	6411	50	6506	
G48	3000	6000	4897	6000	5729	2	5733	50	5926	6000
G49	3000	6000	4903	6000	5709	3	5714	50	5924	6000

Name	Problem Size		Simple Randomized or Randomized-1	Simple Greedy or Greedy-1	Semi-greedy-1 α = 0.5	Simple Local or Local-1		Grasp-1		Known Best Solution or Upper Bound
	|V| or n	|E| or m				Number of Iterations	Best Value	Number of Iterations	Best Value	
G50	3000	6000	4899	5880	5695	3	5700	50	5856	5988
G51	1000	5909	3595	3697	3683	33	3725	50	3745	
G52	1000	5916	3600	3701	3688	33	3731	50	3751	
G53	1000	5914	3603	3700	3684	31	3723	50	3743	
G54	1000	5916	3602	3696	3684	31	3722	50	3746	

The table represents the value calculated for all 54 graphs using random, greedy, semi-greedy, local-search and grasp techniques. The first column consists of the graph number. Second & third column represent the problem size i,e number of vertices and number of edges. Next three columns represent the average value that we got from randomized, greedy and semi-greedy techniques. The next two columns represent iterations required for local search and average local search value subsequently. Furthermore, there are two columns to represent the number of iteration for grasp and the grasp value. The final column represents the best known max-cut values for these graphs.
The above mentioned syntax is followed for each table.

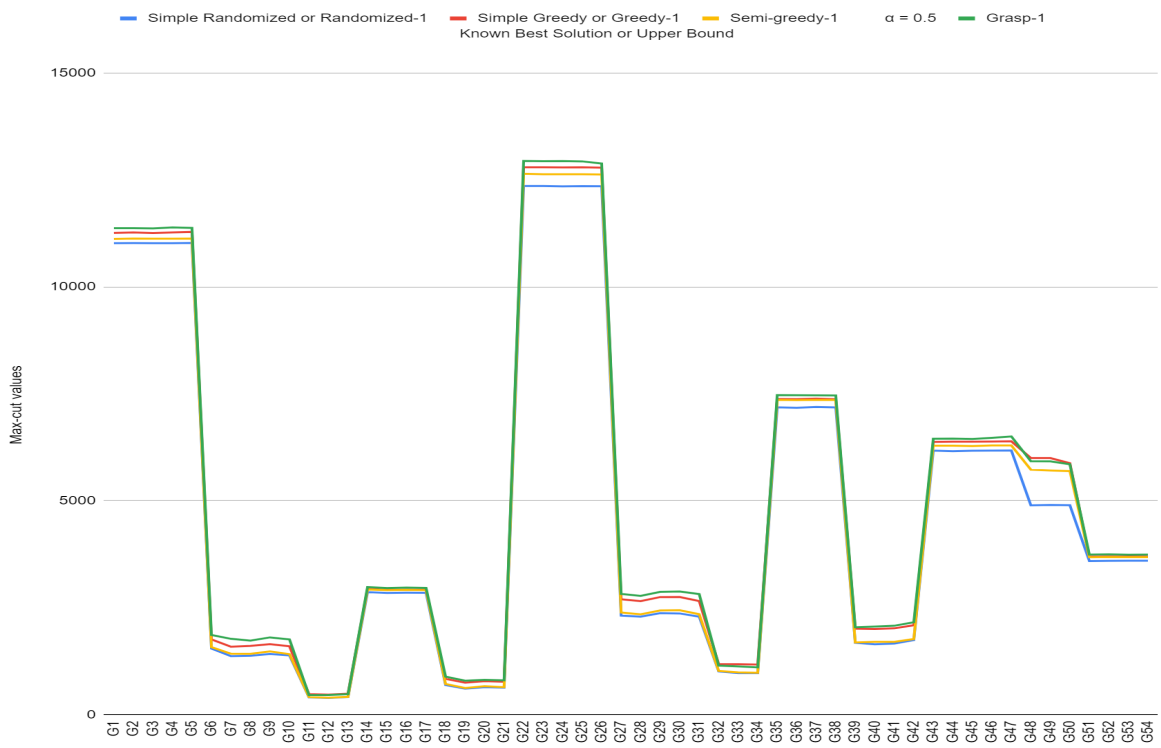
The semi-greedy algorithm follows a value based method to build RCL as per the specification. **α is taken to be 0.5.**

For the randomized algorithm the same function ‘ConstructSemiGreedySolution’ was used with **α=0**. As we know, $\mu = w_{min} + \alpha * (w_{max} - w_{min})$ and the restricted candidate list consists of all the vertices whose value of the greedy function(in this case summation of order) is greater than or equal to μ. Choosing α to be 0 we get, $\mu = w_{min}$. W_{min} is the minimum summation of order among all the vertices. Therefore, the greedy function of every vertice is greater than or equal to μ and they get included in RCL. From the RCL a vertice is chosen at random. As such the function works like a simple randomized algorithm to calculate max-cut.

Similarly, for the greedy algorithm **α is taken 1**, so that **μ=w_{max}** and only the vertices that have the highest summation of order enter RCL. There can be more than one such vertice, so I have taken srand(0) in the main function, so that every time the rand function generates the same output and the same greedy value is generated each time.

After the construction phase the solution from the semi-greedy technique is fed to the 'LocalSearch' function, which implements the first-improvement method to search for a better solution in the neighborhood space. The local search stops when no improving neighbor is found after evaluation of all possible moves (we are stuck in a local optima) which is shown by the number of iterations. The best value of local search is the average of all the local search values for each iteration.

In GRASP the best solution is stored from the construction and the local search phase. As there is randomness introduced in the construction phase, different solution states can get built in each iteration, and so we store the best solution in the grasp method. We kept the number of iterations, **iter=50** for this table.



Graph 1 represents the data from table 1. From the results and the graph it is evident that the average max-cut value calculated from the randomized algorithm is the worst, greedy the best and semi-greedy being in between for all 54 test cases.

It can also be seen that the solution obtained from grasp is much higher than the average, semi-greedy construction, randomized construction.

For **51 out of 54** test cases (excluding g50,g51 & g52) (94.44%) grasp gives better results than the average greedy solution as well. The reason lies in that while constructing greedy solution we opt for local optima, which may not give us the global optima in every case whereas the grasp algorithm is designed to make more than one iteration to first construct a feasible solution then apply iterative improvements until a locally optimum solution is found and store the best locally optimum value from all the iterations.

For graphs **50,50 & 52** (5.56%), we see that the average greedy value gives better results than grasp. It can happen because grasp only stores the best value in all the iterations, whereas we have taken the average of 50 greedy values. The average can be more than an individual value.
Yet the greedy solutions can only reach or go near the known best value for **3/24** test cases (g50, g51 & g52)(12.5%). Because these graphs have more vertices than edges. Max order for each vertex is constant & taking them one by one gives us the best results.

Using Different Constructive Methods:

In the previous table we have calculated the average of randomized, greedy and semi greedy algorithms and compared it with grasp. For the constructive phase of grasp we had only used semi-greedy constructed solutions with $\alpha = 0.5$**.** In this section, we have constructed random, semi-greedy and greedy solutions by individually using local search on each one of them to get the best value or grasp solution. Our motive was to compare between grasp values using 3 different constructive techniques.

1) Randomized Construction:

Table 2:

Name	Problem Size		Simple Randomized or Randomized-1	Simple Local or Local-1		Grasp-1		Known Best Solution or Upper Bound
	|V| or n	|E| or m		Number of Iterations	Best Value	Number of Iterations	Best Value	
G1	800	19176	11021	90	11265	50	11345	12078
G2	800	19176	11027	91	11268	50	11337	12084
G3	800	19176	11020	90	11265	50	11331	12077
G4	800	19176	11037	92	11285	50	11341	
G5	800	19176	11021	91	11260	50	11360	
G6	800	19176	1548	94	1807	50	1895	
G7	800	19176	1391	92	1639	50	1707	
G8	800	19176	1374	97	1646	50	1733	
G9	800	19176	1426	94	1683	50	1769	
G10	800	19176	1376	94	1635	50	1711	
G11	800	1600	407	16	439	50	462	627
G12	800	1600	398	16	430	50	454	621
G13	800	1600	419	19	456	50	474	645
G14	800	4694	2866	48	2933	50	2958	3187
G15	800	4661	2845	49	2913	50	2940	3169
G16	800	4672	2853	46	2917	50	2939	3172
G17	800	4667	2849	50	2917	50	2945	
G18	800	4694	697	69	815	50	856	
G19	800	4661	616	71	739	50	791	
G20	800	4672	647	70	769	50	813	
G21	800	4667	638	72	762	50	816	
G22	2000	19990	12348	189	12728	50	12844	14123
G23	2000	19990	12349	183	12718	50	12809	14129
G24	2000	19990	12363	184	12730	50	12810	14131
G25	2000	19990	12355	186	12731	50	12813	
G26	2000	19990	12349	185	12724	50	12817	
G27	2000	19990	2317	191	2708	50	2779	
G28	2000	19990	2278	195	2676	50	2793	
G29	2000	19990	2382	195	2778	50	2874	
G30	2000	19990	2393	196	2790	50	2874	

Name	Problem Size		Simple Randomized or Randomized-1	Simple Local or Local-1		Grasp-1		Known Best Solution or Upper Bound
	|V| or n	|E| or m		Number of Iterations	Best Value	Number of Iterations	Best Value	
G31	2000	19990	2285	197	2690	50	2791	
G32	2000	4000	1013	42	1098	50	1138	1560
G33	2000	4000	981	44	1070	50	1112	1537
G34	2000	4000	974	44	1064	50	1096	1541
G35	2000	11778	7185	120	7351	50	7394	8000
G36	2000	11766	7178	118	7342	50	7378	7996
G37	2000	11785	7191	120	7358	50	7402	8009
G38	2000	11779	7183	120	7354	50	7391	
G39	2000	11778	1681	174	1985	50	2043	
G40	2000	11766	1648	175	1955	50	2014	
G41	2000	11785	1667	171	1969	50	2022	
G42	2000	11779	1731	180	2046	50	2107	
G43	1000	9990	6170	92	6353	50	6410	7027
G44	1000	9990	6170	96	6359	50	6407	7022
G45	1000	9990	6172	91	6355	50	6422	7020
G46	1000	9990	6166	93	6353	50	6413	
G47	1000	9990	6171	93	6359	50	6444	
G48	3000	6000	4902	76	5077	50	5154	6000
G49	3000	6000	4896	74	5068	50	5170	6000
G50	3000	6000	4899	75	5075	50	5148	5988
G51	1000	5909	3599	59	3681	50	3707	
G52	1000	5916	3602	60	3685	50	3726	
G53	1000	5914	3602	60	3686	50	3719	
G54	1000	5916	3605	58	3686	50	3716	

In Table 2, we have presented the results achieved from **randomized constructive phase**. As mentioned above, we have taken α **= 0** to achieve randomized solution. On the constructed max-cut solution we then did a local search to get the best value, which is presented as the best value in the grasp column. The number of local search iterations and average local search value is also shown.

2) Semi-Greedy (α = 0.5) :

Table 3:

Name	Problem Size		Semi-greedy-1 α = 0.5	Simple Local or Local-1		Grasp-1		Known Best Solution or Upper Bound
	|V| or n	|E| or m		Number of Iterations	Best Value	Number of Iterations	Best Value	
G1	800	19176	11120	73	11302	50	11371	12078
G2	800	19176	11130	71	11311	50	11371	12084
G3	800	19176	11125	69	11299	50	11365	12077
G4	800	19176	11125	71	11310	50	11388	
G5	800	19176	11130	71	11309	50	11378	
G6	800	19176	1582	84	1806	50	1867	
G7	800	19176	1429	83	1658	50	1777	
G8	800	19176	1427	85	1658	50	1736	
G9	800	19176	1482	82	1703	50	1810	
G10	800	19176	1419	84	1650	50	1764	
G11	800	1600	414	15	445	50	462	627
G12	800	1600	401	16	432	50	464	621
G13	800	1600	420	17	455	50	484	645
G14	800	4694	2929	27	2963	50	2986	3187
G15	800	4661	2913	25	2945	50	2964	3169
G16	800	4672	2920	25	2952	50	2974	3172
G17	800	4667	2913	27	2947	50	2965	
G18	800	4694	721	63	830	50	894	
G19	800	4661	628	66	741	50	798	
G20	800	4672	670	62	775	50	817	
G21	800	4667	648	68	763	50	808	
G22	2000	19990	12642	113	12854	50	12943	14123
G23	2000	19990	12631	115	12850	50	12936	14129
G24	2000	19990	12630	118	12853	50	12940	14131
G25	2000	19990	12630	117	12853	50	12932	
G26	2000	19990	12625	111	12831	50	12882	
G27	2000	19990	2393	175	2746	50	2826	
G28	2000	19990	2350	173	2700	50	2780	
G29	2000	19990	2439	174	2790	50	2873	
G30	2000	19990	2445	176	2796	50	2883	
G31	2000	19990	2352	180	2717	50	2822	

Name	Problem Size		Semi-greedy-1 α = 0.5	Simple Local or Local-1		Grasp-1		Known Best Solution or Upper Bound
	|V| or n	|E| or m		Number of Iterations	Best Value	Number of Iterations	Best Value	
G32	2000	4000	1028	38	1104	50	1152	1560
G33	2000	4000	992	41	1076	50	1132	1537
G34	2000	4000	984	42	1068	50	1110	1541
G35	2000	11778	7360	61	7437	50	7471	8000
G36	2000	11766	7353	60	7431	50	7468	7996
G37	2000	11785	7360	61	7437	50	7465	8009
G38	2000	11779	7356	62	7435	50	7464	
G39	2000	11778	1693	170	1984	50	2047	
G40	2000	11766	1707	161	1983	50	2064	
G41	2000	11785	1707	161	1988	50	2082	
G42	2000	11779	1771	167	2063	50	2165	
G43	1000	9990	6289	62	6406	50	6452	7027
G44	1000	9990	6288	60	6402	50	6454	7022
G45	1000	9990	6280	62	6399	50	6445	7020
G46	1000	9990	6294	61	6409	50	6472	
G47	1000	9990	6295	63	6411	50	6506	
G48	3000	6000	5729	2	5733	50	5926	6000
G49	3000	6000	5709	3	5714	50	5924	6000
G50	3000	6000	5695	3	5700	50	5856	5988
G51	1000	5909	3683	33	3725	50	3745	
G52	1000	5916	3688	33	3731	50	3751	
G53	1000	5914	3684	31	3723	50	3743	
G54	1000	5916	3684	31	3722	50	3746	

In Table 3, we have presented the results achieved from a **semi-greedy constructive phase**. As mentioned above, we have taken α **= 0.5**. On the constructed max-cut solution we then did a local search to get the best value, which is presented as the best value in the grasp column. The number of local search iterations and average local search value is also shown.

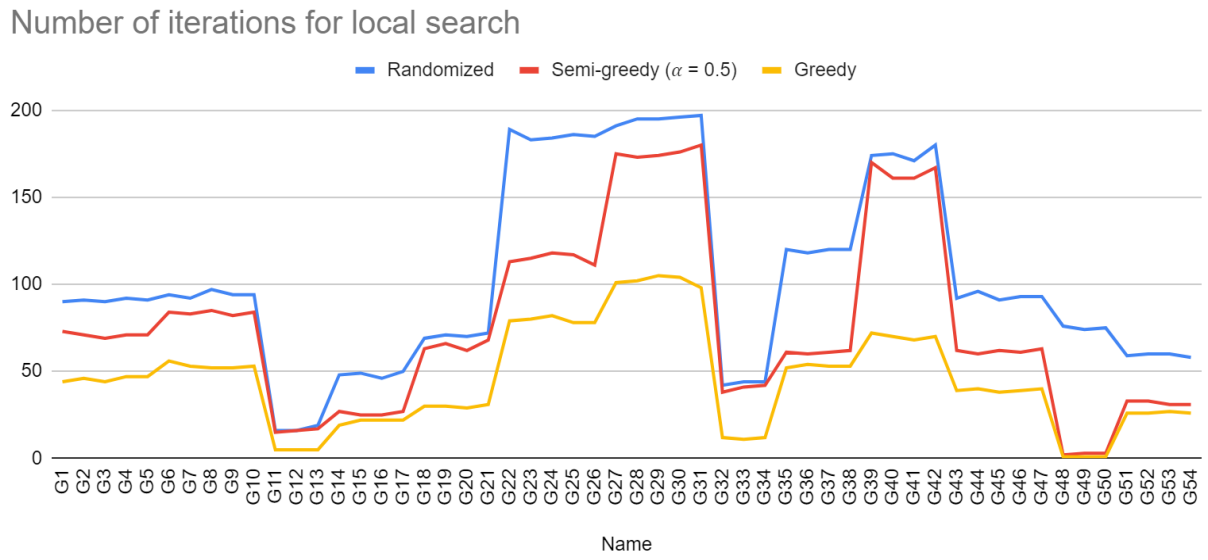
3) Greedy Constructive Method:

Table 4:

Name	Problem Size		Simple Greedy or Greedy-1	Simple Local or Local-1		Grasp-1		Known Best Solution or Upper Bound
	|V| or n	|E| or m		Number of Iterations	Best Value	Number of Iterations	Best Value	
G1	800	19176	11259	44	11368	50	11434	12078
G2	800	19176	11259	46	11366	50	11435	12084
G3	800	19176	11261	44	11372	50	11439	12077
G4	800	19176	11276	47	11388	50	11470	
G5	800	19176	11262	47	11378	50	11443	
G6	800	19176	1767	56	1898	50	1960	
G7	800	19176	1596	53	1727	50	1790	
G8	800	19176	1614	52	1747	50	1805	
G9	800	19176	1650	52	1770	50	1831	
G10	800	19176	1600	53	1723	50	1787	
G11	800	1600	483	5	492	50	504	627
G12	800	1600	470	5	482	50	502	621
G13	800	1600	497	5	504	50	520	645
G14	800	4694	2946	19	2971	50	2987	3187
G15	800	4661	2923	22	2948	50	2969	3169
G16	800	4672	2929	22	2954	50	2972	3172
G17	800	4667	2924	22	2953	50	2968	
G18	800	4694	838	30	881	50	919	
G19	800	4661	757	30	801	50	827	
G20	800	4672	780	29	824	50	872	
G21	800	4667	774	31	815	50	842	
G22	2000	19990	12790	79	12935	50	13007	14123
G23	2000	19990	12800	80	12936	50	13018	14129
G24	2000	19990	12787	82	12928	50	13003	14131
G25	2000	19990	12802	78	12931	50	13000	
G26	2000	19990	12790	78	12925	50	12988	
G27	2000	19990	2691	101	2877	50	2968	
G28	2000	19990	2649	102	2840	50	2927	
G29	2000	19990	2750	105	2938	50	3016	
G30	2000	19990	2756	104	2945	50	3046	
G31	2000	19990	2663	98	2855	50	2931	
G32	2000	4000	1195	12	1215	50	1248	1560

Name	Problem Size		Simple Greedy or Greedy-1	Simple Local or Local-1		Grasp-1		Known Best Solution or Upper Bound
	|V| or n	|E| or m		Number of Iterations	Best Value	Number of Iterations	Best Value	
G33	2000	4000	1180	11	1204	50	1234	1537
G34	2000	4000	1175	12	1201	50	1228	1541
G35	2000	11778	7377	52	7446	50	7473	8000
G36	2000	11766	7373	54	7439	50	7474	7996
G37	2000	11785	7384	53	7448	50	7491	8009
G38	2000	11779	7376	53	7446	50	7484	
G39	2000	11778	2025	72	2126	50	2189	
G40	2000	11766	2002	70	2119	50	2177	
G41	2000	11785	2016	68	2126	50	2187	
G42	2000	11779	2091	70	2192	50	2254	
G43	1000	9990	6384	39	6458	50	6518	7027
G44	1000	9990	6383	40	6450	50	6502	7022
G45	1000	9990	6390	38	6453	50	6496	7020
G46	1000	9990	6386	39	6453	50	6502	
G47	1000	9990	6390	40	6459	50	6511	
G48	3000	6000	6000	1	6000	50	6000	6000
G49	3000	6000	6000	1	6000	50	6000	6000
G50	3000	6000	5880	1	5880	50	5880	5988
G51	1000	5909	3696	26	3729	50	3756	
G52	1000	5916	3706	26	3734	50	3768	
G53	1000	5914	3699	27	3730	50	3757	
G54	1000	5916	3698	26	3728	50	3757	

In Table 4, we have presented the results achieved from the **greedy constructive phase**. As mentioned above, we have taken α **= 1** to achieve greedy solution. On the constructed max-cut solution we then did a local search to get the best value, which is presented as the best value in the grasp column. The number of local search iterations and average local search value is also shown.



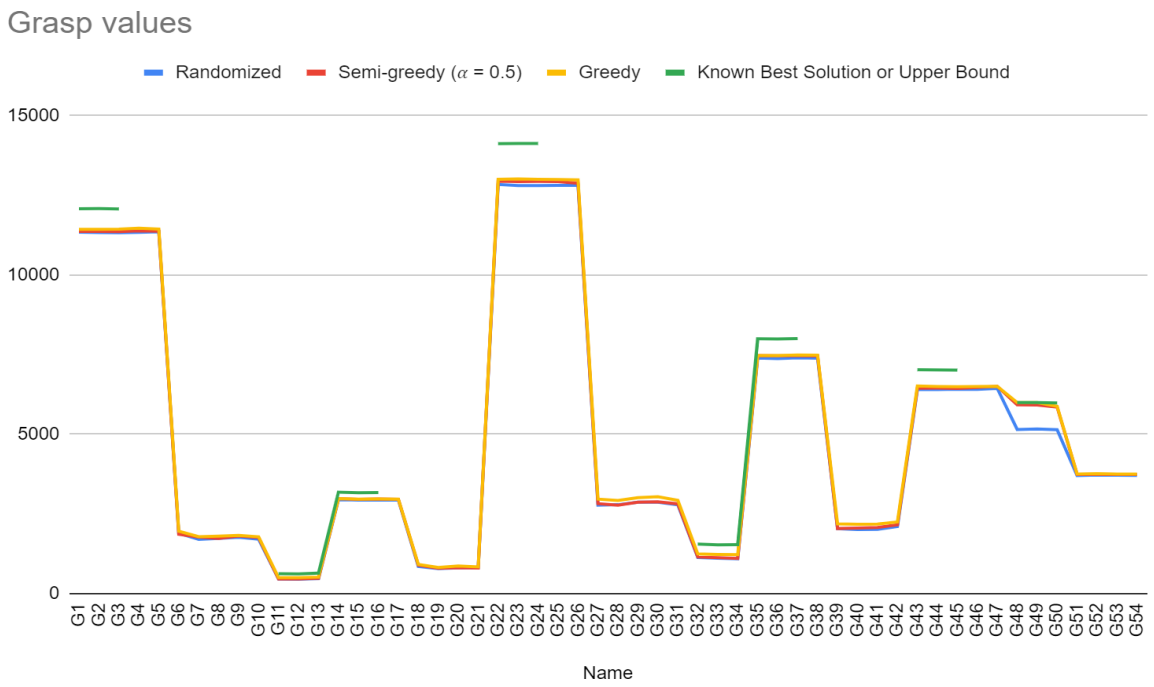
Graph 2 represents the number of iterations required in local search for randomized, semi-greedy and greedy algorithms. The values are taken from **table 2,3 & 4**.

It is evident that the number of iterations in the case of greedy constructive method is the lowest, because whale constructing greedy solutions already take the vertices which have the maximum order sequentially which in turn gives a higher max cut and there's not much scope of randomness. Hence in a local search evaluation not much improvement can be achieved.

The number of iterations required for randomized construction methods is highest as we build a random solution which is certain to give much better improvement in local search.

What is notable is that in graphs **11,12,13,32,33,34 (6/54 = 11.11%**) the number of iterations in semi-greedy methods is near randomized. Because in graphs 11,12,13,32,33 & 34 all of the vertices have the same order. Hence while choosing from them to construct RCL, it works as a randomized method.

Graph **48, 49 & 50 (3/54 = 5.56%**) give number of iteration in case of semi-greedy alike greedy. Because these graphs are sparse. There are half the number of edges than vertices. Hence many vertice have order 0 and thus are chosen in the last for rcl, making the method a greedy one.



Graph 3 represents the grasp values using randomized, semi-greedy and greedy algorithms. The values are taken from **table 2,3 & 4**.

From the graph it is clear that grasp is a very effective method to solve np-hard problems. For 6 out of 24 graphs **11,12,13,48,49 & 50** the grasp best value is equal or very near the known best value.The effectives for our test case results up to **25%.**

The constructed solution for both greedy and semi-greedy gave the same best values. The constructed solution from the randomized algorithm also worked very well apart from **5 graphs out of 54** (9.26%) graphs namely, **g47, g48, g49, g50 & g51**. In case of these graphs most vertices have the same order taking random vertices from the start stocks the search at a local optima. Even though the number of iterations in local search were quite hgh for all of these graphs compared to semi-greedy and greedy, yet a global optimum could not be achieved.